Biophysically accurate floating point neuroprocessors for reconfigurable logic

Yiwei Zhang, Joseph P. McGeehan, *Member IEEE*, Edward M. Regan, Stephen Kelly, Jose Nunez-Yanez, *Member IEEE*

Abstract—**This paper presents a high-performance and biophysically accurate neuroprocessor architecture based on floating point arithmetic and compartmental modeling. It aims to overcome the limitations of traditional hardware neuron models that simplify the required arithmetic using fixed-point models. This can result in arbitrary loss of precision due to rounding errors and data truncation. On the other hand, a neuroprocessor based on a floating-point bio-inspired model, such as the one presented in this work, is able to capture additional cell properties and accurately mimic cellular behaviors required in many neuroscience experiments. The architecture is prototyped in reconfigurable logic obtaining a flexible and adaptable cell and network structure together with real time performance by using the available floating point hardware resources in parallel. The paper also demonstrates model scalability by combining the basic processor components that describe the soma, dendrite and synapse of organic cells to form more complex neuron structures.**

***Index Terms*— bio-inspired model, reconfigurable logic, floating point, neuroprocessor**

I. Introduction

Neural simulation emulates the electrical processes occurring in the brain and allows researchers to investigate the mechanisms underlying bio-electrical signal generation and complex brain functions. In practice, neural simulators are built on different description levels to mimic the behavior of certain neural structures (i.e. soma, dendrite and synapse). Previous research has mostly focused on approximating the cellular electrical process to a level that only reproduces spike-shaped electrical signals which are not biologically meaningful [1, 2]. This simplification results in mismatches between experimental findings and the theoretical models. On the other hand, low-level cell behaviors, such as calcium dynamics and spike modulation, can be reproduced by detailed neuron models [3-5]. These models focus on the description of chemical reactions and electrical processes occurring across the cell membrane, containing biological information at the ionic level. It has been generally accepted that the detailed membrane properties ignored by simplified models play a significant role in generating intricate cell dynamics [3]. These findings suggest that accurate cell models might be the only way of obtaining high correlations between theoretical and empirical findings, although a set of coupled differential equations are involved as a result of low-level modeling. High-precision biophysically detailed models are not commonly used in hardware platforms due to their complex mathematical expressions which required computational intensive floating point arithmetic. To overcome these issues, this paper presents, to the best of our knowledge, the first reconfigurable neuroprocessor architecture based on floating point arithmetic that offers biophysically accuracy and high performance. The contributions of this paper are:

1. A biophysically accurate neuroprocessor architecture in reconfigurable logic using kinetic models and floating point arithmetic. Different cell structures can be emulated on the customized neuroprocessors;
2. An SOC (System-On-Chip) design able to integrate the neuroprocessors and support flexible definition of cell properties and inter-cell connections;
3. An investigation of the postsynaptic effects in a two-cell neural network using the proposed neuroprocessors;
4. The generation and analysis of complex firing patterns using the proposed neuroprocessors.

II. Related Work

Various neural simulators have appeared on different scientific computing platforms over the years [6, 7]. The advent of microelectronics has largely improved computing capability and enabled large population of neurons to be emulated on a single device. Despite this, detailed bio-inspired neuron models have not yet been commonly deployed in existing neural simulators, mainly as a result of the computing performance of the simulators lagging behind the demands of brain emulation. This is important because the ultimate neural simulator is expected to respond to external stimuli by activating

Manuscript received March 6, 2011. This work was supported by Aptina Limited UK and Micron Foundation USA.

Yiwei Zhang, Jose Nunez-Yanez and Joseph P. McGeehan are with the Centre for Communication Research (CCR), University of Bristol, Bristol, BS8 1UB, UK. (e-mail: Y.Zhang@bristol.ac.uk; J.L.Nunez-Yanez@bristol.ac.uk; J.P.McGeehan@bristol.ac.uk).

Edward M. Regan, Stephen Kelly., are with Henry Wellcome L.I.N.E., Clinical Sciences South Bristol, University of Bristol, UK. S. Kelly is also with the New Jersey Neuroscience Institute, JFK Medical Center, New Jersey, USA. (e-mail: er5963@bristol.ac.uk; Stephen.Kelly@bristol.ac.uk).

Digital Object Indentifier 10.1109/TC.2011.257

appropriate biological functions within a biological time scale.

General-purpose processors and DSPs (Digital Signal Processors) have been applied to neuron modeling due to their flexibility. On such platforms, the model is defined in software by using certain standard programming languages (e.g. BRAIN (Python) [7]) or self-script languages (e.g. NEURON [6] and GENESIS [8]). The morphology of the cell or network can be freely tailored if the mathematical model can be solved correctly. For software-designed simulators, the most challenging issue is how to achieve real time performance, which is constrained by limited parallelism available in traditional processor architectures. Thereby, real time simulation is still hardly achieved, even on parallel processors (e.g. PGENESIS [9] and IBM C2 [10]). These systems suffer from low performance, taking minutes to simulate one biological second on tens or hundreds of processors [11].

Artificial neurons have also been fabricated in silicon using analogue techniques, based on modern VLSI (Very Large Scale Integration) technology. It is possible to implement biologically detailed models (calcium dynamics [12], Hodgkin-Huxley (HH) model [13]) or compact a network of simple models (Integrate-and-fire model [1], Izhikevich model [14]) on a single chip. This approach outperforms other simulators because of its lower power consumption (several pJ/spike). Nevertheless, analogue VLSI designs suffer from long design cycles and considerably high costs. It is also difficult to reconfigure and modify the circuit, once the chip fabrication is finished. Finally, the noise resulting from transistor mismatches, can give rise to imprecision and unexpected responses.

Another alternative is GPGPU computing (General Purpose Graphic Processing Unit), which is regarded as a novel solution to neural simulation [15, 16]. The abstract model which contains a large number of dot product operations can be efficiently implemented on this platform. In this case, the GPU-based simulator can achieve 18 to 24 times improvement [17], compared with the Pentium platform. Therefore, many researchers have diverted to this method to develop the artificial cells, such as the work reported in [17]. The latest research shows the fastest GPGPU (the Tesla S1070) can achieve up to 84.4 times of speedup over the quadcore 2.67 GHz Xeon Processor [18]. However, the neuron was modeled as a single compartment without any morphological information. It is important to note that the GPU is not well-suited for conditional branching and iterative operations because of its parallel hardware nature. In this case, its computing performance may be less than traditional CPUs [19]. In particular, GPUs have not been reported as the simulation platform for multi-compartment modeling. It has also been pointed out that the memory bandwidth is another main limitation in the GPU, which causes memory conflict and stalling stages during the simulation [17].

Neural simulators have also been designed on state-of-the-art FPGAs, which offer highly parallel programmable interconnects and massive logic resources. The existing FPGA-based simulators focus on the network modeling based on abstract models (single-point model [20], Izhikevich model [11]) or biologically detailed models (conductance-based model [21]). On FPGAs, it is possible to emulate cell behaviors at each modeling level in biological real time, as well as supporting reconfigurable design [22]. However, numerical precision has focused almost exclusively on fixed point arithmetic. Although this simplification yields smaller circuit sizes, our previous research has demonstrated that the position of the spike is shifted and the spike amplitude is truncated if fixed point representation is chosen [23]. The effects of these anomalies in brain functionality are not fully understood and this has motivated us to target floating point arithmetic in this work. Furthermore, bio-inspired models such as HH are not commonly used in FPGA-based simulators, although it has been suggested that the HH-type model is the only class of model which is capable of delivering biophysically meaningful information [24].

Our previous research [23] has demonstrated a biophysically accurate floating point somatic neuroprocessor based on the HH model. It has been shown that this somatic processor is able to reproduce identical electrical signals compared with the original experimental model. The proposed system was developed based on parallel FPALUs (Floating Point Arithmetic Logic Units) and FSMCs (Finite State Machine Controllers). In addition, it has been demonstrated that the effect of cumulative round-off error introduced by fixed point representation leads to a shift of spikes along the time axis, which potentially affects brain function in an error-prone way. For example, if the neuron is modeled using 32-bit Q15 format (32-bit fixed point number with 15-bit integer), the fifth spike may occur ahead of 4 ms in comparison with the 32-bit floating point solution. The work proposed in this paper extends our previous work including the Traub soma, active/passive dendrites and kinetic-based synapse models. As will be shown later, these additional components are required to generate more complex cell firing patterns and synaptic connections among cells. The proposed neuroprocessor architecture is modular so that it can be used as a building block in a complex neural simulator, capable of emulating the functionality of different sections of the brain. In the complete system, all the design considerations, i.e. biophysical accuracy, numerical precision, system reconfigurability and real time performance are taken into account. The architecture takes advantage of the dedicated hardwired computing components available in modern FPGAs, the kinetic-based neuroprocessors use 32-bit floating point arithmetic and are able to capture the cell characteristics at the ionic level within a biological time scale. The entire system is developed around an SOC (System-On-Chip) architecture to add support for reconfigurable inter-cell connectivity and individual cell property definition.

III. Mathematical Model Overview

This section presents the reference neuron models used in this research, including the HH model, the Traub somatic, dendritic model and the kinetic-based synaptic model. The flexibility of

the model is increased with the use of compartments that allow the synaptic connections to be located at different positions along the dendritic tree. The introduction of multiple compartments may result in a numerically stiff system which should be solved by using explicit numerical methods.

A. Bio-inspired Cell Model

The bio-inspired model focuses on the mathematical description of ion channel conductances which are determined by the ion exchange across the cell membrane. It is capable of producing biophysically meaningful information [24], although it is mathematically complex. The HH model [4], as one of the most successful cell models, explains the contribution of sodium and potassium channels to the membrane voltage change with respect to the time variable. Similarly, Traub proposed a more refined kinetic-based neuron model by adding the calcium dynamics to the dendrite [3]. In these two models, each ionic conductance is described by the product of the peak conductance and channel state variables which follow the kinetic rule as given in Equations (1) and (2):

$$i_{ion} = \overline{g}_{\max} \cdot \gamma_1^{p_1} \cdot \gamma_2^{p_2} \cdots \gamma_n^{p_n} \qquad (1)$$

$$\frac{d\gamma_i}{dt} = \alpha_i \cdot [T] \cdot (1-\gamma_i) - \beta_i \cdot \gamma_i \qquad (2)$$

where i_{ion} is the ionic channel current; γ_i represents the active or inactive state variable of ionic channels; p_i defines the rate of state variable change; α_i and β_i are defined as the forward and backward rates, respectively. *[T]* stands for the concentration of reactants, having its peak value 1 mill-Mole. For both neuron models, α and β have similar definitions as given in Equations (3) and (4):

$$s1 = a(V_m - \theta_1) / [\exp(\frac{V_m - \theta_1}{b_1}) + 1] \qquad (3)$$

$$s2 = a \cdot \exp[b_1 \cdot (V_m - \theta_1) + b_2 \cdot (V_m - \theta_2)] \qquad (4)$$

where V_m indicates the membrane voltage; *s1* and *s2* are two forms of the forward or backward rate definitions. *a, b,* b_1, θ_1 and θ_2 are constant parameters depending on the cell model. The decision of which equation to use to define α and β is determined by the ionic channel. For example, in the HH soma model, the forward rate of the sodium inactivation state variable and the backward rates of the sodium and potassium activation state variables are defined by Equation (4), whereas other transfer rates follow Equation (3) [4].

In addition, to mimic the inter-cell spike coupling, synapses are taken into account. The kinetic-based synapse model proposed by Destexhe [25] is used in this work. It outperforms other exponential models [26, 27] or abstract weighting-value models [28], as it uses Equation (2) to describe the chemical transmitter release phenomenon, constructing the link between the chemical reaction and electrical process.

In general, the cell models mentioned above can be generalized to have the form of the product of peak conductance and the kinetic-based state variables whose forward and backward rates are voltage- and time-dependent. For somata or dendrites, the definition of the forward and backward rates involves the sigmoid or exponential function. In the case of the synapse, these two rates are constants. This similarity between cell models makes it possible to design a configurable computing architecture where the data path can be extracted and the control unit is individually devised in terms of different cell models. In doing so, a single architecture is capable of supporting the emulation of the kinetic-based models for somata, dendrites or synapses.

B. Multi-compartment modeling

To include spatial information and apply the incoming spike at different positions along the neuron cable, the multi-compartmental modeling approach is used in this work. In this method, the soma, dendrite and synapse models (compartments) are joined together. Each compartment can be equipped with passive or active ionic channels or chemical transmitters, as well as coupling with one or two adjacent compartments via the cytoplasm conductance. In our system, an arbitrary number of compartments can be connected together to represent a nerve cell, if there are sufficient hardware resources available in the FPGA.

When the multi-compartment approach is applied to the neuron modeling, a group of coupled ordinary differential equations are introduced. The membrane potential of each compartment is computed using Equation (4):

$$C_m \cdot \frac{dV_j}{dt} = i_e - i_m - i_{syn} + \frac{a}{2r_a} \cdot \frac{V_{j+1} - 2V_j + V_{j-1}}{(\Delta x)^2} \qquad (5)$$

where r_a and C_m are the specific cytoplasm resistance, respectively; Δx is the spatial grid; a is the radius of the compartment; V_{j-1}, V_j and V_{j+1} *(j=0,1,..,L)* are the membrane voltages of three adjacent compartments and L is the total number of compartments. i_e, i_m and i_{syn} denote the input, ionic and synaptic currents per unit area, respectively. The length of spatial grid Δx should be smaller than 0.1λ, where λ is the electrical length of the cell [8]. Within each Δx-length compartment, the cell has unique properties and Equation (5) can be defined with two or three unknown voltages. To solve the multi-compartment model, appropriate numerical methods are required. In this research, we have employed exponential and backward Euler methods to maintain system stability instead of explicit forward Euler or Runge-Kutta methods [29]. These two explicit methods can result in considerably erroneous membrane voltages with an increasing number of compartments. When the system becomes unstable, their resulting membrane voltages demonstrate oscillating waveforms even with small integration steps (0.1 ms). For example, in an 8-compartment model with the HH soma and passive dendrite, a 0.5 μA/cm^2 stimulating current can cause an oscillating membrane voltage in the case of explicit solutions. As a semi-implicit method, the exponential Euler can keep the solution stable in many cases, therefore it is used as the default integration method in the neuroprocessor. However, under certain values of the cell properties or stimuli (e.g. the quantity of compartments, ion channel conductances, cytoplasmic conductance, cell geometric

properties, stimulating current amplitude, etc.), its solution also gradually starts oscillating. Therefore, according to the register setting of the neuroprocessor, the backward Euler can be used to implicitly integrate cell models with second-order accuracy, when the individual cell contains several compartments. The downside of implicit methods is increasing computational overhead. For example, an Integrated-and-fire model can be solved within 5-step computations, whereas there are 27 additions/subtractions, 25 multiplications, 13 divisions and 10 exponentials executed during each time step in the case of a single-compartment HH model with the exponential Euler solution. If the backward Euler method is applied, the numbers of basic arithmetic computations are 26, 25, 12 and 9, respectively. Therefore, it is a challenging issue of solving the model with implicit methods in biological real time.

IV. FLOATING POINT NEUROPROCESSORS

Following the previous work published in [23] that presented the HH neuroprocessor, the architecture is extended with Traub somatic, dendritic and synaptic neuroprocessors, to generate more complex spiking patterns and enable the construction of synaptic connections among the cells. Given the spike shift and amplitude truncation introduced by fixed point algorithms [23], these neuroprocessors are devised in IEEE-754 32-bit floating point data format. This is important because the timing of spikes is believed to carry crucial information among cells. Nevertheless, floating point computation leads to a considerably large silicon area, if the circuit is conventionally designed using logic resources. To offset this limitation, we have investigated a new type of scalable floating point neuroprocessor architecture in [23], taking advantage of the dedicated computing resources available in the modern FPGAs (DSP48a slices in Xilinx devices) to perform floating point arithmetic with a smaller circuit size. In addition, the neuroprocessor supports up to 4 virtual cells to be individually modeled on each neuroprocessor.

A. Configurable neuroprocessor architecture

The HH somatic, Traub somatic, Traub dendritic and synaptic processors are based on a configurable control and data path architecture as illustrated in Fig.1.

(Fig. 1.)

The control and data paths are formed by a number of configurable FSMCs and FPALUs respectively. The FSMCs are devised to fetch operands or forward computation results between the FPALUs and their associated RAMs. All temporary variables are saved in these internal memories that are located at the input ports of the FPALU. The data paths between the FPALU and RAMs are manipulated by the FSMCs. All the FSMCs control the sequence of computation in cooperation with each other, based on a predefined state order. This state order is devised to make full use of the FPALUs to improve the computation efficiency, taking into consideration some scenarios where the operands cannot be forwarded to the FPALUs in time. In this way, all the FPALUs operate in parallel and are almost fully occupied during the computation with a small number of idle states. By redefining the computation order in the FSMCs, this common architecture can perform the emulation of different neuron structures. For example, Fig.2 demonstrates how the states transits in the adder of the dendritic processor, in which active/passive dendrite and two Euler methods are taken into consideration. This transition definition controls the operation of the adder and its two associated internal RAMs, targeting full occupation of the adder. In the active dendrite mode, the additions/subtractions defined in the cell model are executed in S0 to S23 states, yielding the total ionic conductance and current. Alternatively, the adder directly begins its computation from S23 in the passive dendrite mode. Afterwards, the computation is performed, following the order defined in S24_exp to S30_exp in the exponential Euler method or S25_back to S28_back for the backward Euler solution. The result of the adder is stored in different internal RAMs, which will be used in the later operations.

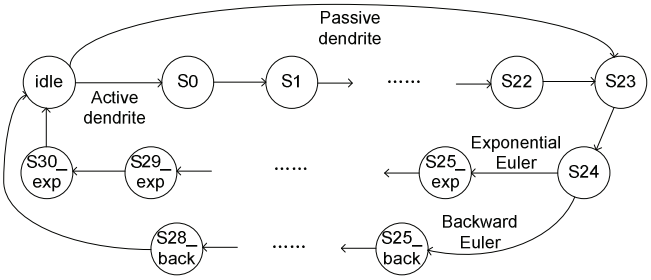


Fig. 2. State machine transition graph of the FSMC.

Regarding the FPALUs, their architectures are more complex than fixed point ALUs. Fig.3 shows the architecture for the adder/subtractor and multiplier/divider FPALUs. The addition and subtraction involve the operations of exponent comparison and increment/decrement, mantissa shift and result normalisation. The multiplication and division are performed by the operations of exponent addition/subtraction, mantissa multiplication/division and result normalisation. In particular, the multiplication can be realized by using the available DSP48a slice on the current platform (Spartan3aDSP_1800a). The DSP48a is a dedicated component on Xilinx FPGAs to facilitate mathematical functions in an efficient way. Fig.4 depicts the architecture of the DSP48a-based multiplier. As the internal multiplier in the DSP48a slice has 18-bit data bus, 4 DSP48a slices are needed to perform 24x24 bit mantissa multiplication. After passing through the internal multipliers and post-adders of the DSP48a, the results of four data paths are shifted and aligned to obtain the resulting mantissa. For multiplication, the exponent and sign can be obtained by addition and XOR logic operations, as shown in Fig.3.(b).

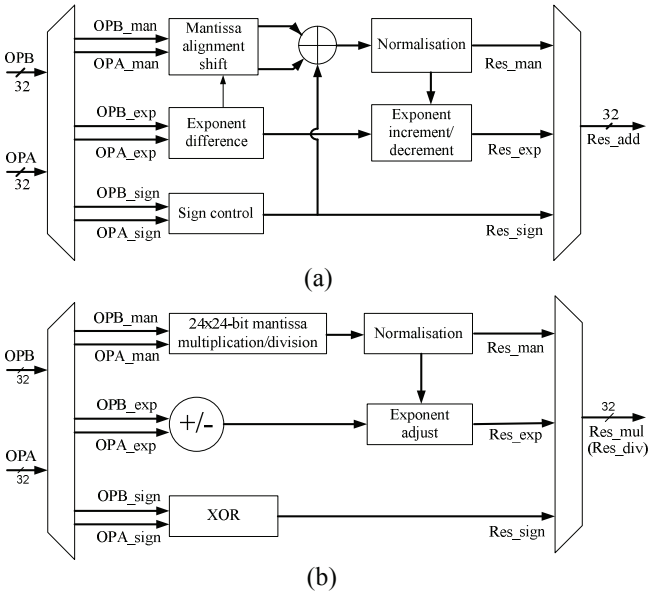


Fig. 3. Architecture of FPALUs. (a) Floating point adder/subtractor; (b) Floating point multiplier/divider.

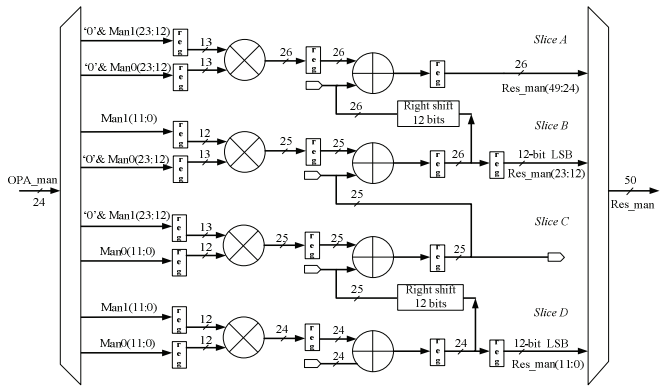


Fig. 4. Architecture of DSP48a-based floating point mantissa multiplier, which is constructed according to [30].

Fig.5 illustrates the architecture of the floating point comparator, which compares the two input signs, exponents or mantissas in sequence. Only if the previous comparison obtains the equal result, should the next step be executed. Fig.6 is the basic architecture of the floating point exponential ALU, which has been developed based on the CORDIC algorithm in our previous research [23]. The input exponent is pre-processed to guarantee it locates in the convergence range of the CORDIC algorithm. The CORDIC iteration is performed in the CORDIC processing block, mainly containing one look-up table, two floating point adders and several multiplexers. At the final stage, the result is adjusted according to the pre-processing output. Such architecture performs the exponential very accurately as evaluated in [23].

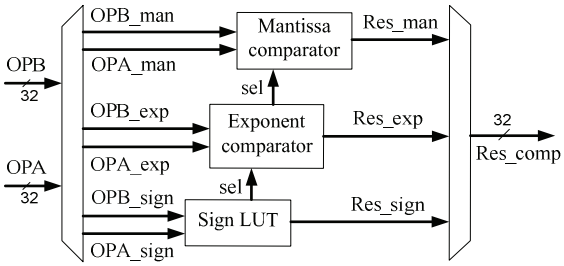


Fig. 5. Architecture of floating point comparator.

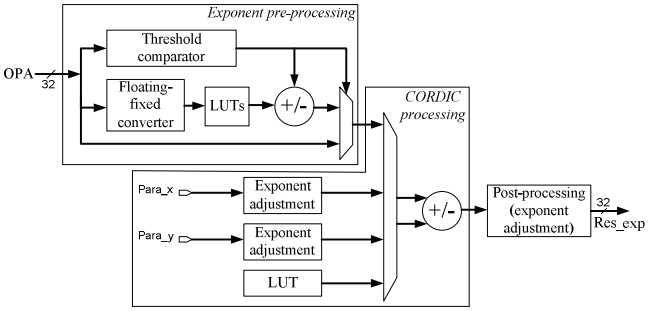


Fig. 6. Architecture of floating point CORDIC exponential.

The neuroprocessor communicates with the system bus via a DPRAM (Dual-Port RAM), the 'parameter-and-result memory', in which all the cell parameters and results are stored, such as the equilibrium potential and maximal conductance of ion channels, membrane capacitance, cytoplasm resistance, geometric properties, etc. For each neuroprocessor, which support up to 4 virtual cell models, this RAM uses 256 words and 32 bits per word. The first 8 addresses are used as register space to store operation setting for all virtual cells, including Euler method, multiplexing mode, etc. Following these settings, the neuroprocessor supports up to 54 parameters plus 8 configuration registers for each virtual cell. This memory space can be freely read or written by the neuroprocessor or the embedded main processor. Therefore, it is possible to investigate the cell behaviors by using a variety of cell properties on the neuroprocessor. Table 1 shows the storage format of this RAM.

(Table 1)

For the emulation of the soma or the dendrite, according to the register setting, the neuroprocessors compute the membrane voltage when the exponential Euler is chosen to integrate the differential equations. If the backward Euler is chosen to solve multi-compartment cells, a matrix equation is involved. In this case, each neuroprocessor outputs four matrix coefficients and send them to the main processor where all the coefficients are collected together and the voltage is finally obtained. If the neuroprocessor is used to mimic the synaptic behavior, the synaptic current and conductance are computed, rather than the membrane voltage. At the end of the simulation step, the computation results are written back to this DPRAM, which can be taken by the main embedded processor to display the results on the terminal window or solve the matrix equation in the

backward Euler mode.

The DSP48a slice, which is the on-chip dedicated high performance computing component, is applied to perform 24x24 bit mantissa multiplication for the floating point multiplier. Such design reduces the number of lookup tables and flip-flops to 178 and 245 respectively, compared with the 646 lookup tables and 703 flip-flops used in the logic-only strategy. This dedicated computing unit can also be deployed to construct floating point adder, reducing lookup tables from 424 to 246 and flip-flops from 485 to 300. For the floating point exponential solver, a dedicated circuit has been developed to avoid multiplication and division, which has been proposed in our previous work [23]. This exponential solver is devised based on the CORDIC algorithm. It is capable of processing arbitrary input exponents within the neural dynamic range without convergence limit issues and the iteration step can be freely defined with each iteration. It has been shown in [23] that the error is zero when the input is located within the processing range. Other FPALUs are generated by configuring floating point cores provided by ISE Coregen tool. Table 2 summarizes the logic utilisation on the current Spartan3aDSP1800a FPGA platform. Apart from multiplier, the DSP48 slice also supports floating point adder/subtractor in the Virtex FPGA series. As the FSMCs are designed to avoid low utilization of the FPALUs, all the FPALUs are almost fully occupied with a small number of idle states.

Table 2. Logic utilisation and core latency of FPALUs.

ALU	LUT	Flip-flop	DSP48a	Latency
Adder	574	410	-	8 cycles
Multiplier	178	245	4	8 cycles
Divider	827	1,367	-	26 cycles
CORDIC exp	2,264	1,961	-	170 cycles
Fix2float	230	226	-	3 cycles
Float2fix	269	232	-	3 cycles
comparator	91	19	-	3 cycles

In the current design, arbitrary values can be used as the integral time step Δt according to the definition in the 'parameter-and-result' memory. The simulations given herein were obtained by using 0.1 ms time step, because 10 KHz sampling rate is frequently used in biological experiments, which is regarded as the biological real time scale.

B. The Traub somatic processor

The Traub somatic processor performs the computation of the membrane voltage defined in the Traub soma model, including the sodium and delay-rectified potassium active channels. The calcium dynamics is included in the dendritic processor, rather than the somatic processor. This is because the calcium and calcium-mediated channels are mainly presented in distal compartments in the Traub model [3] and are removed from the somatic part in the reduced Traub model proposed in [31]. This neuroprocessor contains four basic FPALUs and it is based on the configurable architecture. It was designed by replacing the FSMCs in the HH processor. The new FSMC organizes the computational order in an efficient way, in terms of the Traub soma model. In the Traub somatic processor, the data bus width to the ALU are 5, 5, 4 and 4 bits, which are labeled as A1 and A2 (addition), B1 and B2 (multiplication), C1 and C2 (division), and D1 (exponential) in Fig.1.

The computation latency of the Traub neuroprocessor is about 2,000 cycles, achieving similar performance to the HH neuroprocessor (about 2,200 cycles/output) [23]. As a result, this neuroprocessor takes 20 μs to update the membrane voltage at 100 MHz system clock frequency, supporting up to 4 virtual somata in multiplexing mode. The design was synthesized and implemented using SynplifyPro 9.6.2 and ISE 11.3 toolset. Its logic utilisation is given in Table 3.

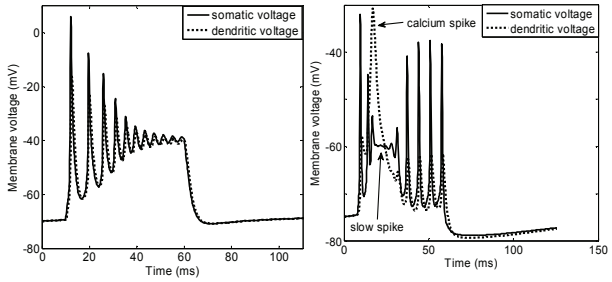
C. Passive/active dendritic processor

The dendritic processor is developed to emulate the activity of the Traub active dendrite or passive dendrite. The Traub active dendrite [31], contains a group of calcium-dependent channels, exerting nonlinear effects on the membrane potential, whereas the passive dendrite is modeled by a constant resistance which only attenuates the membrane voltage along the dendrite.

The dendritic processor is devised based on the common architecture illustrated in Fig.1. In addition to the four basic FPALUs, a floating point comparator is employed. This comparator performs the membrane potential and calcium level comparison with the threshold values as defined in the Traub dendrite model [3]. The design of the FSMCs is changed according to the mathematical definition of the model, as well as supporting the configurable passive or active model selection. Its data bus width to the ALU are 5 (A1 and A2), 5 (B1 and B2), 3 (C1 and C2), 3 (D1) and 1 (E1 and E2) bits. Furthermore, the dendritic processor provides the wire connection to the synaptic processor to obtain the postsynaptic conductance and current during each time step. In doing so, the synaptic connection can be located at an arbitrary position along the dendritic cable.

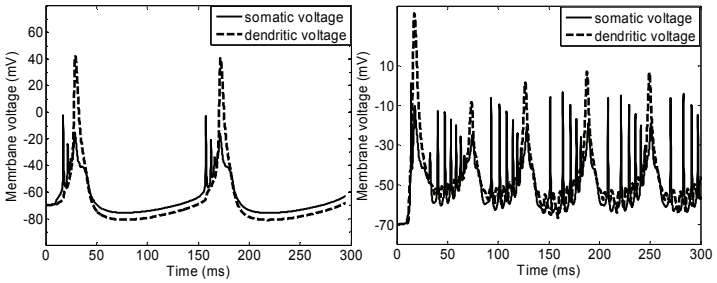
It takes around 19.2 μs in the active Traub dendrite mode if the system clock is 100 MHz (equivalent to 1,920 cycles). The computation latency for the passive dendrite model is 3.54 μs under the same clock rate (equivalent to 354 cycles). Both scenarios satisfy the requirement of biological real time, supporting up to 4 virtual models in one processor. Its logic utilisation is shown in Table 3.

To clarify the significance of the active dendrite processor, a simulation was run. Fig.7 compares the membrane voltage of the soma and dendrite in the case of passive dendrite and active Traub dendrite. It can be seen that the passive dendrite only attenuates the membrane voltage along the cable, whereas the active one exerts nonlinear effect on the signal. The calcium spike and slow spike are generated at the dendrite and soma, respectively. And the voltage tends to be more negative after the burst. These phenomenon cannot be delivered by the passive dendrite, although such firing activities carry important information between cells.



(a) (b)

Fig. 7 The effect of active dendrite processor. (a) passive dendrite is attched; (b) active Traub dendrite is attached.



(a) (b)

Fig. 8 The effect of synaptic processor.(a) no synaptic connections between two cells; (b) synapstic connections are set up between two cells.

D. Synaptic processor

The synaptic neuroprocessor emulates the behavior of three types of chemical synapses: excitatory AMPA, NMDA and inhibitory GABAa receptors. All chemical receptors are modeled by using the kinetic formula presented in [32]. As the synapses are associated with the dendrite, this processor computes the total postsynaptic conductance and current instead of the membrane potential, when it receives the incoming spike from the source neuron. Its neighboring dendritic voltage is used as the compartmental voltage in this processor. At the end of the computation, the synaptic processor forwards the results to the connecting dendritic processor, exerting the postsynaptic effect on the dendrite. Moreover, in real neurons, a second action potential cannot be induced instantaneously after the previous one. Given this fact, the synaptic processor ignores the successive incoming spikes within a time window (i.e. dead time) which is defined in its 'parameter-and-result memory'. The population of each type of synapse can be assigned at the beginning of each time step. It is also possible to disable any type of synapse by simply setting its peak conductance or quantity to zero. In doing so, the excitatory or inhibitory synaptic effect can be individually emulated on the processor.

Like the somatic and dendritic processors, this architecture is also applicable to the synaptic processor. It contains five types of basic ALUs and associated FSMCs and internal memory. The floating point comparator is also employed for dead time comparison. In this processor, the data bus width to the ALU are 5 (A1 and A2), 6 or 5 (B1 and B2), 3 (C1 and C2), 1 (D1) and 3 (E1 and E2). Its computation latency is around 15.5 μs with a 100 MHz clock (equivalent to 1,550 cycles), supporting up to 4 virtual synaptic compartments. Table 3 gives its logic usage summary.

Fig.8 demonstrates the effect of synaptic connections. In Fig.8.(a), the stimulating current at the soma caused the action potentials. The burst in Fig.8.(b) is also evoked by the coupling synaptic current. In comparison with the case of two individual cells, the neuron is excitated more frequently and delivers more complex information, when the two cells are couple d via synaptic connections. Therefore, keeping this information in the neural simulator plays a significant role in neuron behavior investigation.

V. SOC ARCHITECTURE AND SOFTWARE ENVIRONMENT

This section presents the SOC architecture where the proposed neuroprocessors are combined together, taking into account the spatial information. Based on this structure, a software environment has been developed for system configuration and data processing. On the current platform, the final stage is a DAC board where the resulting voltage is converted to analogue waveforms. The system has been prototype in an FPGA board equipped with a Xilinx XC3SD1800A device. This board has been chosen because a DAC daughter board is available which is required in the following experiments for easy extraction of the output membrane voltage waveforms.

A. Leon3 and AMBA system

To be able to parameterise the neuroprocessors and connect the artificial cells, an embedded microprocessor, the Leon3 processor, is integrated into this design, leading to an SOC architecture. As the central control unit, the Leon3 configures and triggers the computation of the neuroprocessors, as well as dispatching action potentials between cells in place of using wire connections. To integrate the neuroprocessors and this global processor, AMBA (Advanced Microcontroller Bus Architecture) is involved to attach all modules together, as shown in Fig.9, In addition, a floating point coprocessor (GRFPU) is employed to assist the Leon3 to perform post-data processing in the backward Euler mode. Furthermore, to synchronise all the neuroprocessors in the neural network, a hardware module, 'nnwsynch_pro', has been developed. The function of this module is to send a '1' to the GPIO (General-Purpose I/O) module when the computation of all neuroprocessors is complete at each simulation time step to avoid occupying multiple interrupt lines. Then this high bit is transferred to the interrupt controller to generate and forward an interrupt to the Leon3 processor. When the Leon3 receives this interrupt, application software will be executed to process the resulting data.

(Fig. 9) and (Table 3)

As the neuroprocessors run faster than the biological real time scale, resource sharing is supported. Fig.10 demonstrates

system computation scheduling when the neuroprocessors work in the multiplexing mode. Each neuroprocessor can support up to 4 virtual cells within a single configurable time step. The virtual cell can be routed to other cells via individual data bus, thereby resource sharing affects the system structure by introducing more connections. At the beginning of the time step, the Leon3 configures and triggers the neuroprocessors, which calculates the membrane voltage or post-synaptic current for each virtual structure. After all the computation is completed, the Leon3 reads and forwards the firing status between local memories under the control of network synchronization module ('nnwsynch_pro'). Alternatively, the spike delivery can be done via direct wire connections between source and target cells.

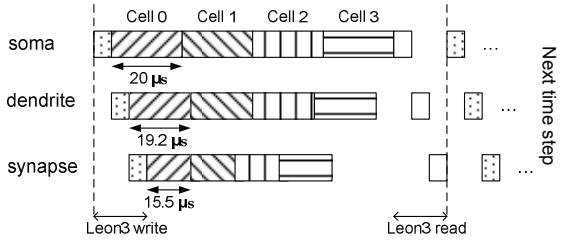


Fig. 10. Multiplexing computation scheduling.

B. *Software environment*

The application software which runs on the Leon3 performs three main functions: system initialisation, interrupt handler and matrix equation solver for the backward Euler method. At the beginning of the simulation, the whole system must be initialised by configuring a group of registers and memories mapped in the 'parameter-and-result memories'. The task of the interrupt handler is to read the computation result of each neuroprocessor and then write a 'start' signal to the neuroprocessors to trigger the computation of the next time step. When the backward Euler method is chosen to solve the multi-compartment model, the Leon3 executes the matrix equation solver function to obtain the membrane potential of compartments based on the coefficients computed by the neuroprocessor. Fig.11 shows the entire experimental setup.

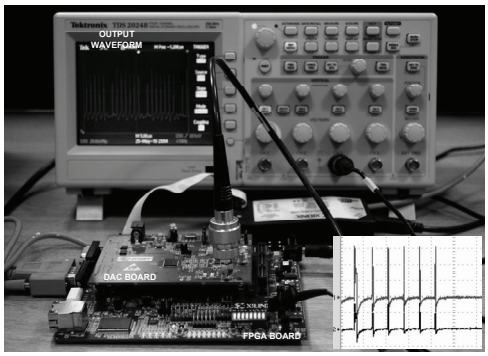


Fig. 11. Photo of the experimental setup, including an analogue board to display outputs and generate stimulus. The resulting membrane voltage is displayed on the oscilloscope.

VI. PERFORMANCE

Experiments were run on neuroprocessors and CPUs for performance analysis. As we are focusing on highly accurate simulation, a large number of neurons are not simulated at the current stage, which will be considered in the future research. Fig.12 demonstrates the performance improvement of the neuroprocessor. When operating at 100 MHz clock, the HH somatic neuroprocessor can achieve about 30, 19 and 12 times of speedups over the Pentium IV 1.8G/1GB, 2.8G/1GB and 3G/3GB processors, respectively. Similar results were obtained on other neuroprocessors. As the computation latency of the neuroprocessor is fixed, its performance can be further improved by clocking the FPGA at higher frequency.

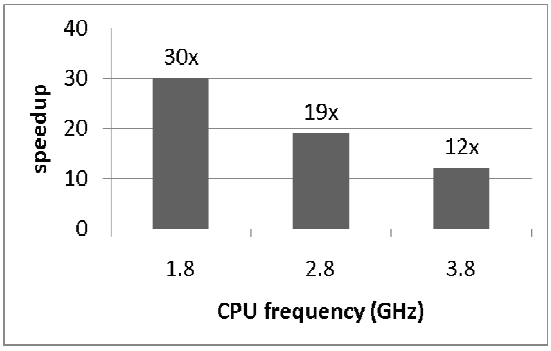


Fig. 12. Performance improvement of the neuroprocessor.

As inter-cell spikes can be alternatively delivered via wire connection or register access, heavy traffic on the bus can be avoided by using physical connections. This is a trade-off between performance and circuit size. In addition, as parallel computing and multiplexing scheme are supported by the neuroprocessors, its performance scale can linearly increase in the exponential Euler mode. For the backward Euler method, the linear speedup may break down in the case of a large number of neurons. This is because the Leon3 will limit the overall performance. To address this issue, processors with higher performance than Leon3 can be introduced, which will be studied in our future research.

VII. EXPERIMENTAL RESULTS

To evaluate the neuroprocessors, a group of simulations were run to mimic the behavior of a small network and individual cell with different cell properties and stimuli. Most of cell parameters are taken from the original data in [3, 4, 25].

A. *A 2-cell neural network*

A small neural network including 2 neurons was tested using the neuroprocessors, each of which, as illustrated in Fig.13, is extended from the soma to the terminating dendrite through three dendritic compartments. The chemical synapses are attached on the distal dendrites of the two neurons. The somatic processor sends its spikes to the synaptic processor of another cell. In this simulation, the somata have sphere shapes with diameters of 86.6 µm (*Neuron0*) and 20.0 µm (*Neuron1*), respectively. For *Neuron0*, the diameters of its three cylindrical

dendrites are 5.0 μm, 5.0 μm and 2.4 μm and the lengths are 11.0 μm, 11.0 μm and 20.0 μm, respectively. For *Neuron1*, the diameters of its three dendrites are 12.0 μm, 12.0 μm and 8.0 μm and the lengths are 15.0 μm, 15.0 μm and 10.0 μm, respectively. The population of chemical receptors per 100 μm^2 of cell membrane are estimated as: 60 (AMPA), 60 (NMDA) and 10 (GABAa) [33]. Other parameters of the cell property are presented by Traub in [3].

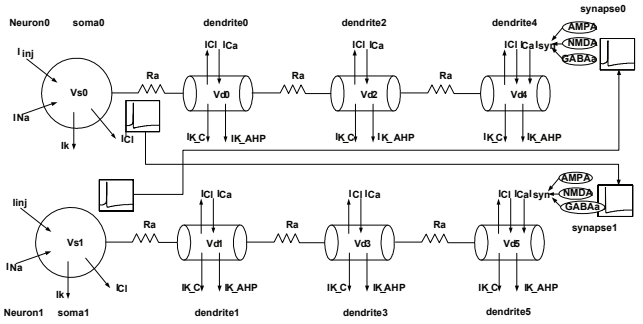
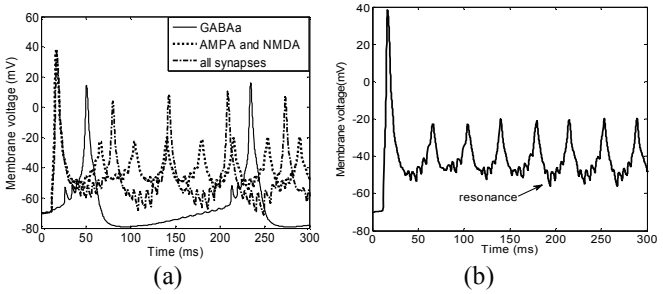


Fig. 13. Interconnections between neuroprocessors to emulate a 2-neuron neural network.

Fig.14 shows the output waveform of the 2-cell neural network with different type of synaptic connections between the two cells. The inhibitory impact of the GABAa synapse can be observed by disabling the AMPA and NMDA synapses in the synaptic neuroprocessor. As shown in Fig.14.(a), in comparison with the excitatory effect, the spike train was postponed due to the inhibitory effect of GABAa synapses. It also demonstrates the excitatory postsynaptic effect reproduced by the neuroprocessors. In this test, the AMPA and NMDA receptors were enabled. Comparing those curves in Fig.14.(a), it is evident that the two neurons were firing more frequently as the result of excitatory postsynaptic currents. It is also shown that the nerve cells associated with both excitatory and inhibitory synapses were firing with higher rate than the inhibitory effect case, but with lower frequency than the excitatory postsynaptic effect scenario. Moreover, the subthreshold oscillating phenomenon (i.e. resonance) was also generated with synaptic connections, as shown in Fig.14.(b) Such cell burst patterns reproduced by these synaptic connection have also been found in the literature [5, 21, 34].



(a) (b)

Fig. 14 Simulation results of the 2-cell network. (a) various synaptic effects; (b) resonance bursting.

B. Burst pattern analysis

As the neuroprocessors are developed based on the empirical recordings, they are capable of replicating a variety of burst patterns. The diversity of burst patterns plays a significant role in understanding the mechanism of signal processing inside the brain.

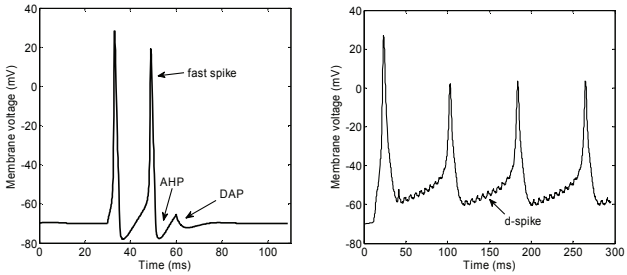
B.1 Depolarising afterpotential (DAP) and Afterpotential Hyperpolarising (AHP)

In the first test, we connected the Hodgkin-Huxley somatic processor with the passive dendritic processor. As demonstrated in Fig.15.(a), a burst including two fast spikes was induced and terminated with an AHP and a following DAP. Such AHP and DAP phenomena have been recorded experimentally in [34, 35]. The passive dendrite does not have any impact on action potential, except for voltage attenuation.

B.2 D-spikes

Based on *in vitro* experimental results, Schwartzkroin [5, 34] defined the d-spike as the fast spike with 2-15 mV amplitude and 8-9 ms falling edge. These characteristics were generated on the neuroprocessor as shown in Fig.15.(b).

In this experiment, the d-spike was observed at the proximal dendrite, preceding the normal spike. It was not reproduced at the somatic part, since it is normally remote from the soma [34]. Moreover, as shown in Fig.8.(b), the postsynaptic effect can also elicit d-spikes riding on the membrane potential at the proximal and distal dendrites.



(a) (b)

Fig. 15. Various bursting patterns generated by the neuroprocessors. (a) fast spikes, AHP and DAP; (b) d-spikes.

B.3 Calcium dynamics on the soma and active dendrites

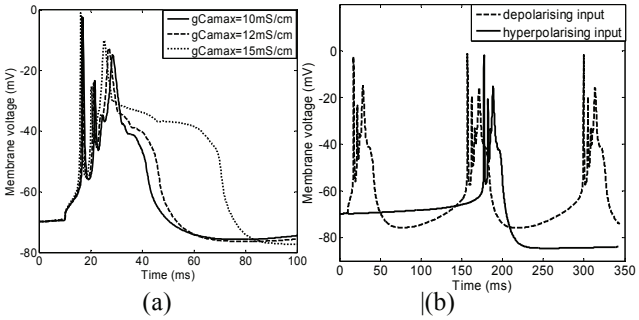
The calcium dynamics is one of most significant mechanisms by directly affecting the neuron or activating calcium-dependent ionic currents. It has been duplicated on the Traub somatic and dendritic neuroprocessors, as shown in Fig.7.(b). A burst, which consists of fast spikes and calcium-mediated spikes, was reproduced on the somatic and dendritic processors. The normal fast spikes are followed by successive slow spikes as the result of the calcium activity in the dendrite [34]. A large calcium spike is evoked at the dendrite with larger amplitude and longer duration than the fast spike. When the calcium spike occurs in the dendrite, the soma demonstrates interposed slow spikes as response to the dendritic calcium dynamics. The AHP was also reproduced in this model with prolonged durations. All these

phenomena are consistent with the experimental measurement [35, 36].

To explore the effect of the conductance of calcium channels, another group of simulations were run by applying different peak conductances to the calcium channel. As shown in Fig.16.(a), the somatic membrane voltages were obtained using three peak calcium conductances: 10 mS/cm^2 (solid line), 12 mS/cm^2 (thick dashed line) and 15 mS/cm^2 (thin dashed line). It can be seen that the prolonged spike is driven by the active calcium system and it sustains longer time in the case of larger calcium conductance.

B.4 Cell burst evoked by the hyperpolarising currents

The hyperpolarising current flows across the cell membrane, leading to more negative membrane voltage, thus the action potential tends to be inhibited by this current. In Fig.16.(b), the solid line is the resulting membrane potential at the soma, where a hyperpolarising current was applied. The dashed line is the membrane potential in the case of a depolarising current input, which has the same amplitude with the hyperpolarising one. A spike delay occurs with the hyperpolarising input current. This phenomenon has also been observed in biological experiments [34].

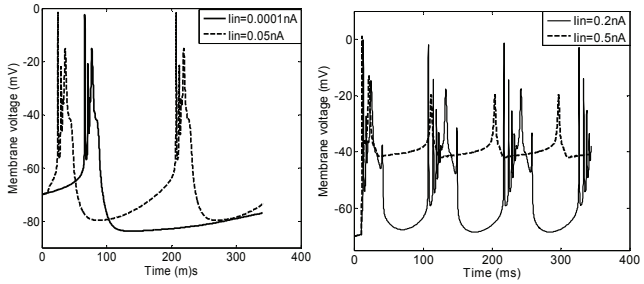


(a) |(b)

Fig. 16. Simulation waveforms of the Traub somatic processor. (a) effect of the calcium channel conductance; (b) effect of the hyperpolarising current

B.5 Cell burst induced by depolarising currents

The depolarising current is the current which makes the membrane potential more positive or less negative with respect to the surrounding environment. When it is applied to the neuron, different burst patterns can be generated, as given in Fig.17. As the stimulating current increases, repetitive bursts occur. In Fig.17.(a), it is demonstrated that the cell is firing in a relatively stable way, when the injected current is sufficiently small. In addition, it has also been shown in Fig.17.(b) that the repetitive rate of the burst train and the quantity of action potentials in a burst are on the increase with larger input currents. It is also worth mentioning that the cell returns to the steady status immediately after a large calcium spike (Fig.17.(a)) or successive fast spikes (Fig.17.(b)). Long lasting AHP and rebound bursts were also generated and the burst is blocked with sufficiently strong inputs (Fig.17.(b)). All these phenomena have been recorded experimentally [34, 36, 37].



(a) |(b)

Fig. 17. Bursts patterns excited by a long steady depolarising step current on the Traub soma. (a) Iin=0.0001nA and Iin=0.05nA; (b) Iin=0.2nA and Iin=0.5nA.

From the experiments shown herein, it can be concluded that some significant cell burst patterns can be replicated using the neuroprocessors. For these artificial output phenomena, the corresponding experimental records have been observed in biological measurements. It should be noted that it is not possible to generate such firing patterns by using simplified models. As the biologically relevant details are removed from those models, they are unable to deliver the accurate information which can be interpreted by the real brain.

VIII. CONCLUSION AND FUTURE WORK

We have presented four types of reconfigurable biophysically accurate floating point neuroprocessors (HH somatic, Traub somatic, dendritic and synaptic neuroprocessors), based on a configurable control and data path architecture. The behaviors of the main structures of living neurons are emulated at ionic level within the biological time scale. Both single and multiple compartment models are supported by this platform, which can be solved by the exponential or backward Euler methods to maintain system stability. The floating point solution eliminates the round-off and truncation error introduced by the fixed point algorithm. The resulting membrane voltage can be converted into analogue signals ready to interface with the organic brain.

The paper also demonstrates the ability of the hardware to reproduce a variety of biologically accurate firing patterns. These firing phenomena were replicated by constructing artificial neurons with different cell properties, stimulating currents or inter-cell connections. In doing so, a series of burst patterns are obtained, including depolarising afterpotential, repetitive burst and fast spike, interpolated slow spike, large calcium spike, d-spike, burst frequency modulation by the stimulating current, suppressed burst modulated by a large stimulating current and spike delay with the hyperpolarising current. All these burst patterns have the corresponding observation recordings obtained from biological experiments.

At the current stage, we are interested in emulation accuracy, rather than the scale of networks. As a result, each single neuroprocessor is capable of delivering ionic-level firing information in a biophysically and numerically accurate way. In our future research, many artificial neurons will be networked

on larger and multiple FPGA chips, in which each cell can be tailored using a group of neuroprocessors. Future work also involves extending the presented hardware with interface capabilities to organic cells culture in-vitro. This will lead us towards the development of a bio-electronic brain that combines biological and artificial neurons in a single computing platform. This system could yield significant further insights in the field of neuroscience research.

REFERENCES

1. Indiveri, G., E. Chicca, and R. Douglas, *A VLSI array of low-power spiking neurons and bistable synapses with spike-timing dependent plasticity.* IEEE Transactions on Neural Networks, 2006. 17(1): p. 211-221.
2. Izhikevich, E.M., *Simple model of spiking neurons.* IEEE Transactions on Neural Networks, 2003. 14(6): p. 1569-1572.
3. Traub, R.D., et al., *A model of a CA3 hippocampal pyramidal neuron incorporating voltage-clamp data on intrinsic conductances.* Journal of Neurophysiology, 1991. 66(2): p. 635-650.
4. Hodgkin, A.L. and A.F. Huxley, *A quantitative description of membrane current and its application to conduction and excitation in nerve.* Journal of Physiology, 1952. 117: p. 500-544.
5. Schwartzkroin, P.A., *Characteristics of CA1 neurons recorded intracellularly in hippocampal in vitro slice preparation.* Brain Research, 1975. 85(3): p. 423-436.
6. Hines, M.L. and N.T. Carnevale, *The NEURON simulation environment.* Neural Computation, 1997. 9(6): p. 1179-1209.
7. Goodman, D. and R. Brette, *Brian: a simulator for spiking neural networks in Python.* Frontiers in Neuroinformatics, 2008.
8. Bower, J.M. and D. Beeman, *The book of GENESIS*. 1998: Springer-Verlag New York, Inc.
9. Klopp, J.C., et al., *Large scale simulations of hippocampal-neocortical interactions in a parallel version of genesis.* Computational Neuroscience: Trends in Research, 1998: p. 59-64.
10. Ananthanarayanan, R. and D. Modha. *Anatomy of a cortical simulator*. in *Proceedings of Supercomputing*. 2007.
11. Cassidy, A. and A.G. Andreou, *Dynamical digital silicon neurons.* 2008 IEEE Biomedical Circuits and Systems Conference - Intelligent Biomedical Systems (Biocas), 2008: p. 289-292.
12. Rasche, C. and R. Douglas, *An improved silicon neuron.* Analog Integrated Circuits and Signal Processing, 2000. 23(3): p. 227-236.
13. Zou, Q., et al., *Real-time simulations of networks of Hodgkin-Huxley neurons using analog circuits.* Neurocomputing, 2006. 69(10-12): p. 1137-1140.
14. Wijekoon, J.H.B. and P. Dudek, *Integrated circuit implementation of a cortical neuron.* Proceedings of 2008 IEEE International Symposium on Circuits and Systems, Vols 1-10, 2008: p. 1784-1787.
15. Fidjeland, A.K., et al., *NeMo: A Platform for Neural Modelling of Spiking Neurons Using GPUs.* 2009 20Th IEEE International Conference on Application-Specific Systems, Architectures and Processors, 2009: p. 137-144.
16. Nageswaran, J.M., et al., *A configurable simulation environment for the efficient simulation of large-scale spiking neural networks on graphics processors.* Neural Networks, 2009. 22(5-6): p. 791-800.
17. Nageswaran, J.M., et al. *Efficient simulation of large-scale spiking neural networks using CUDA graphics processors*. in *International conference on neural networks*. 2009.
18. Taha, T.M. and B. Han, *Acceleration of spiking neural network based pattern recognition on NVIDIA graphics processors.* Applied Optics, 2010. 49(10): p. B83-B91.
19. Meuth, R. and D. Wunsch. *A survey of neural computation on graphics processing hardware*. in *Symposium on Intelligent Control*. 2007.
20. Nedjah, N., et al., *Reconfigurable MAC-based architecture for parallel hardware implementation on FPGAs of artificial neural networks.* Artificial Neural Networks - Icann 2008, Pt Ii, 2008. 5164: p. 169-178.
21. Hughes, S.W., et al., *NeuReal: An interactive simulation system for implementing artificial dendrites and large hybrid networks.* Journal of Neuroscience Methods, 2008. 169(2): p. 290-301.
22. Maguire, L.P., et al., *Challenges for large-scale implementations of spiking neural networks on FPGAs.* Neurocomputing, 2007. 71(1-3): p. 13-29.
23. Zhang, Y., et al. *A biophysically accurate floating point somatic neuroprocessor*. in *The International Conference on Field Programmable Logic and Applications* 2009.
24. Izhikevich, E.M., *Which model to use for cortical spiking neurons?* IEEE Transactions on Neural Networks, 2004. 15(5): p. 1063-1070.
25. Destexhe, A., Z.F. Mainen, and T.J. Sejnowski, *An efficient method for computing synaptic conductances based on a kinetic-model of receptor-binding.* Neural Computation, 1994. 6(1): p. 14-18.
26. Durstewitz, D., J.K. Seamans, and T.J. Sejnowski, *Dopamine-mediated stabilization of delay-period activity in a network model of prefrontal cortex.* Journal of Neurophysiology, 2000. 83(3): p. 1733-1750.
27. Rall, W., *Branching dendritic trees and motoneuron membrane resistivity* Experimental Neurology, 1959. 1(5): p. 491-527.
28. Nedjah, N., et al., *Dynamic MAC-based architecture of artificial neural networks suitable for hardware implementation on FPGAs.* Neurocomputing, 2009. 72(10-12): p. 2171-2179.
29. Mascagni, M.V. and A.S. Sherman, *Numerical methods for neuronal modeling*, in *Methods in Neuronal Modeling: From Ions to Networks*, C. Koch and I. Segev, Editors. 1998, The MIT Press.
30. Xilinx, I. *XtremeDSP DSP48A for Spartan-3A DSP FPGAs User Guide*. 2008 [cited.
31. Pinsky, P.F. and J. Rinzel, *Intrinsic and network rhythmogenesis in a reduced Traub model for Ca3 neurons.* Journal of Computational Neuroscience, 1995. 2(3): p. 275-275.
32. Destexhe, A., Z.F. Mainen, and T.J. Sejnowski, *Kinetic models of synaptic transmission*, in *Methods in Neuronal Modeling: from synapses to networks*, C. Koch and I. Segev, Editors. 1998, the MIT Press, Cambridge.
33. Destexhe, A., et al., *Fluctuating synaptic conductances recreate in vivo-like activity in neocortical neurons.* Neuroscience, 2001. 107(1): p. 13-24.
34. Schwartzkroin, P.A., *Further characteristics of hippocampal CA1 cells In vitro.* Brain Research, 1977. 128(1): p. 53-68.
35. Traub, R.D. and R. Llinas, *Hippocampal pyramidal cells - significance of dendritic ionic conductances for neuronal function and epileptogenesis.* Journal of Neurophysiology, 1979. 42(2): p. 476-496.
36. Traub, R.D., *Simulation of intrinsic bursting in CA3 hippocampal-neurons.* Neuroscience, 1982. 7(5): p. 1233-1242.
37. Wong, R.K.S. and D.A. Prince, *Afterpotential generation in hippocampal pyramidal cell.* Neurophysicol, 1978. 45: p. 86-97.

Yiwei Zhang received the B.Eng. degree from Sichuan University, Chengdu, China, in 2002 and the M.Sc degree from the China Academy of Telecommunications Technology, Beijing, China, in 2005. From 2003 to 2006, she was with the Datang Mobile, Beijing, China, where she became an FPGA and ASIC design and verification engineer. She joined the University of Bristol in 2006. She received her PhD degree at the Centre for Communications Research, University of Bristol, UK in 2011. Her research interests include neuron modelling, neuron sensor, FPGA and ASIC design and verification and SOC design.

Jose Luis Nunez-Yanez received the B.S. degree from Universidad de La Coruna, La Coruna, Spain, and the M.S. degree from Universidad Politécnica de Cataluña, Barcelona, Spain, in 1993 and 1997, respectively, both in electronics engineering, and the Ph.D. degree in the area of hardware architectures for high-speed data compression from Loughborough University, Loughborough, U.K., in 2001.

He is a Research Fellow with the Department of Electronic Engineering, Loughborough University, Loughborough, U.K., where he has worked since 1997. His current research interests include the areas of lossless data compression, reconfigurable vector architectures, FPGA-based design, and high-speed data networks.

Joseph P. McGeehan received the B.Eng. and Ph.D. degrees in electrical and electronic engineering and the D.Eng. degree for his significant contribution to the field of mobile communications research from the University of Liverpool, Liverpool, U.K., in 1967, 1971, and 2003, respectively.

He is currently a Professor of communications engineering and the Director of the Centre for Communications Research, University of Bristol, Bristol, U.K. He is concurrently the Managing Director of Toshiba Research Europe Ltd., U.K. He has been actively researching spectrum-efficient mobile radio communication systems since 1973 and has pioneered work in many areas, including linear modulation, linear power amplifiers, smart antennas, propagation modeling, phaselocked loops and medical electronics.

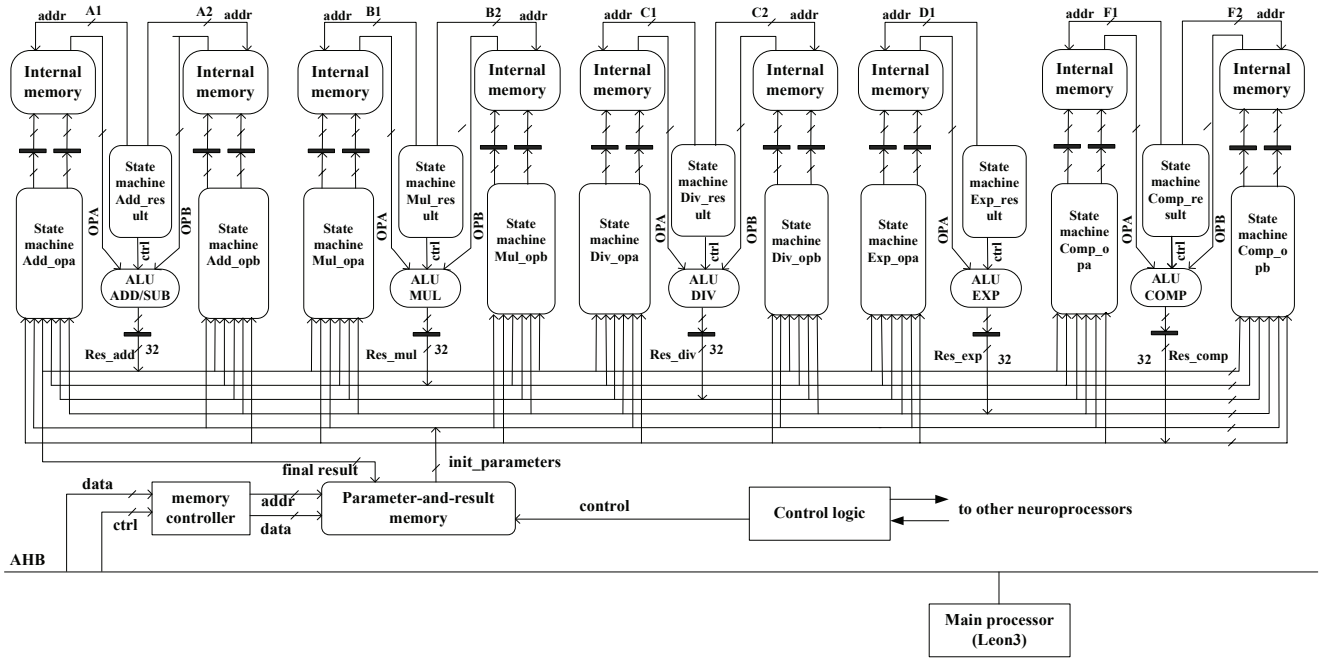


Fig. 1. Architecture of the generic neuroprocessor.

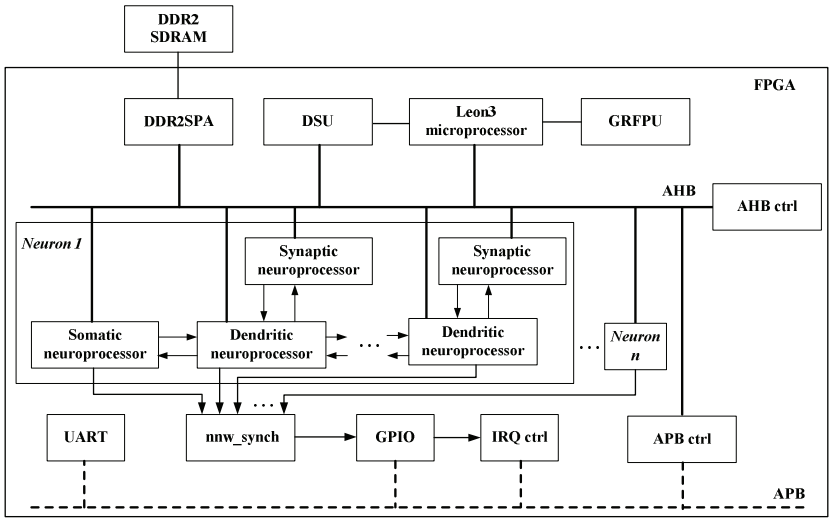


Fig. 9. Schematic of the entire SOC system

Table 1. The parameter-and-result memory.

Address	Register/parameter	Note
0	Start register	Written by the Leon3 to trigger the computation
1	CONFIG1 register	Compartment number, Euler, multiplexcity mode, etc.
…	…	…
8	Virtual cell 1 configuration	Left/right boundary, input current setting
…	…	…
16 ..19	Coefficient_vector 1..4 (result)	Four matrix coefficient for the backward Euler
20	Vm (result)	Output membrane voltage for the exponential Euler
…	…	…
26	Ena	Equilibrium potential of sodium channel
…	…	…
70	Virtual cell 2 configuration	Left/right boundary, input current setting
...	...	...

Table 3. Logic utilisation summary.

xc3sd1800a-4fg676 FPGA	HH soma	Traub soma	Dendrite	Synapse	Leon3	Pre-analogue processing
Slice Flip Flops	4,716 (14.2%)	4,745 (14.3%)	6,705 (20.3%)	6,507 (19.6%)	7,140 (21.5%)	1,332 (4%)
4 input LUTs	6,396 (19.2%)	6,486 (19.5%)	8,252 (24.8%)	6,052 (18.2%)	9,755 (29.3%)	10,822 (32.5%)
Slice Flip Flops (CORDIC ALU)	1,961 (5.9%)	1,961 (5.9%)	1,961 (5.9%)	1,961 (5.9%)	-	-
4 input LUTs (CORDIC ALU)	2,264 (6.8%)	2,264 (6.8%)	2,264 (6.8%)	2,264 (6.8%)	-	-
DSP48a	4 (4.8%)	4 (4.8%)	4 (4.8%)	4 (4.8%)	1 (1.2%)	-
Bits of Block RAMs	28K (1.9%)	28K (1.9%)	27.25K (1.8%)	28.25K (1.9%)	504K (33.3%)	
Computation duration (clock cycles)	2,200	2,000	1,920(a)/354(p)*	1,550	-	-
Performance	up to 130 MHz				40 MHz	Up to 200MHz

* ‘a’ represents active dendrite; ‘p’ means passive dendrite.